

**What Is Claimed Is:**

1           1.       A method to indicate to a dynamic compiler of a multitasking  
2 virtual machine when the dynamic compiler can skip generating native code for a  
3 class initialization barrier when compiling a program method, wherein a runtime  
4 representation of classes by the multitasking virtual machine includes a shared  
5 runtime data structure that is shared by multiple tasks, and wherein a native code  
6 produced by the dynamic compiler extends the shared runtime data structure  
7 representing classes and can be executed serially or concurrently by multiple tasks  
8 of the multitasking virtual machine, the method comprising:  
9           augmenting the shared runtime data structure representing a shared part of  
10 a class with an initializer field; and  
11           using the initializer field of the class to determine whether a platform-  
12 independent instruction of the program method may trigger an initialization of the  
13 class.

1           2.       The method of claim 1, further comprising initializing a bootstrap  
2 class, wherein the bootstrap class is initialized during startup of a task of the  
3 multitasking virtual machine, and before any concurrency, due to creation of  
4 multiple threads of control within the task, takes place.

1           3.       The method of claim 2, further comprising assigning a value of an  
2 initializer of the class when the class is fully initialized, wherein the value  
3 includes an indication that one of:  
4           the class is not the bootstrap class,

5 the class is the bootstrap class and the value of the initializer identifies a  
6 holder of the class initialization barrier that triggered the initialization of the class,  
7 and  
8 the class is the bootstrap class and the value of the initializer further  
9 indicates that the class initialization was not triggered by a class initialization  
10 barrier.

1 4. The method of claim 3, further comprising:  
2 setting a binary variable to zero upon starting the multitasking virtual  
3 machine; and  
4 setting the binary variable to one when all bootstrap classes have been  
5 initialized by a first task executed by the multitasking virtual machine;  
6 whereby the binary variable indicates to the multitasking virtual machine  
7 whether all bootstrap classes have been initialized.

1 5. The method of claim 4, further comprising:  
2 upon initiating the initialization of the class from a class initialization  
3 barrier, noting the holder of the class initialization barrier; and  
4 once the class is fully initialized, assigning the holder to the initializer  
5 field only if the binary variable is zero.

1 6. The method of claim 3, further comprising upon setting a non-  
2 bootstrap class to a fully initialized state for a task, assigning the initializer field  
3 of the class to a constant value, wherein the constant value is distinguishable from  
4 all other possible values for the initializer field.

1           7.     The method of claim 6, wherein the constant value is a NULL  
2     pointer.

1           8.     The method of claim 3, wherein a pointer to a runtime data  
2     structure representing the shared part of the class is assigned to the initializer field  
3     of the class to indicate that the class is the bootstrap class whose initialization is  
4     not triggered by a class initialization barrier.

1           9.     The method of claim 3, further comprising:  
2     if the class does not have an initialization sequence,  
3                 setting the class to a fully initialized state upon the class  
4                 being loading by the task, and  
5                 assigning the initializer field of the class to a pointer to a  
6                 runtime data structure representing the shared part of the class.

1           10.    The method of claim 3, further comprising instructing the dynamic  
2     compiler not to generate native code for the class initialization barrier of the  
3     program method being compiled if the class targeted by the class initialization  
4     barrier is equal to the class that defines the program method being compiled.

1           11.    The method of claim 10, further comprising instructing the  
2     dynamic compiler not to generate native code for the class initialization barrier of  
3     the program method being compiled if the class targeted by the class initialization  
4     barrier is a superclass of the class that defines the program method being  
5     compiled.

1           12.     The method of claim 11, wherein the value of the holder of the  
2     class initialization barrier is a pointer to a runtime data structure representing the  
3     shared part of the class that defines the program method that holds the class  
4     initialization barrier.

1           13.     The method of claim 12, further comprising instructing the  
2     dynamic compiler not to generate native code for the class initialization barrier of  
3     the program method being compiled if the value of the initializer field of the class  
4     targeted by the class initialization barrier is:

5           different from the value that indicates that the class is not bootstrap class,  
6     and

7           different from the pointer to the runtime data structure representing the  
8     shared part of the class that defines the program method being compiled.

1           14.     The method of claim 11, wherein the value of the holder of the  
2     class initialization barrier is a pointer to the shared runtime data structure  
3     representing the program method that holds the class initialization barrier.

1           15.     The method of claim 14, further comprising instructing the  
2     dynamic compiler not to generate native code for the class initialization barrier of  
3     the program method being compiled if the value of the initializer field of the class  
4     targeted by the class initialization barrier is:

5           different from the value that indicate that the class is not the bootstrap  
6     class, and

7           different from the pointer to the shared runtime data structure representing  
8     the program method being compiled.

1           16.     A computer-readable storage medium storing instructions that  
2     when executed by a computer cause the computer to perform a method to indicate  
3     to a dynamic compiler of a multitasking virtual machine when the dynamic  
4     compiler can skip generating native code for a class initialization barrier when  
5     compiling a program method, wherein a runtime representation of classes by the  
6     multitasking virtual machine includes a shared runtime data structure that is  
7     shared by multiple tasks, and wherein a native code produced by the dynamic  
8     compiler extends the shared runtime data structure representing classes and can be  
9     executed serially or concurrently by multiple tasks of the multitasking virtual  
10    machine, the method comprising:

11           augmenting the shared runtime data structure representing a shared part of  
12    a class with an initializer field; and

13           using the initializer field of the class to determine whether a platform-  
14    independent instruction of the program method may trigger an initialization of the  
15    class.

1           17.     The computer-readable storage medium of claim 16, the method  
2     further comprising initializing a bootstrap class, wherein the bootstrap class is  
3     initialized during startup of a task of the multitasking virtual machine, and before  
4     any concurrency, due to creation of multiple threads of control within the task,  
5     takes place.

1           18.     The computer-readable storage medium of claim 17, the method  
2     further comprising assigning a value of an initializer of the class when the class is  
3     fully initialized, wherein the value includes an indication that one of:

4           the class is not the bootstrap class,

5 the class is the bootstrap class and the value of the initializer identifies a  
6 holder of the class initialization barrier that triggered the initialization of the class,  
7 and  
8 the class is the bootstrap class and the value of the initializer further  
9 indicates that the class initialization was not triggered by a class initialization  
10 barrier.

1 19. The computer-readable storage medium of claim 18, the method  
2 further comprising:  
3 setting a binary variable to zero upon starting the multitasking virtual  
4 machine; and  
5 setting the binary variable to one when all bootstrap classes have been  
6 initialized by a first task executed by the multitasking virtual machine;  
7 whereby the binary variable indicates to the multitasking virtual machine  
8 whether all bootstrap classes have been initialized.

1 20. The computer-readable storage medium of claim 19, the method  
2 further comprising:  
3 upon initiating the initialization of the class from a class initialization  
4 barrier, noting the holder of the class initialization barrier; and  
5 once the class is fully initialized, assigning the holder to the initializer  
6 field only if the binary variable is zero.

1 21. The computer-readable storage medium of claim 18, the method  
2 further comprising upon setting a non-bootstrap class to a fully initialized state for  
3 a task, assigning the initializer field of the class to a constant value, wherein the

4 constant value is distinguishable from all other possible values for the initializer  
5 field.

1 22. The computer-readable storage medium of claim 21, wherein the  
2 constant value is a NULL pointer.

1 23. The computer-readable storage medium of claim 18, wherein a  
2 pointer to a runtime data structure representing the shared part of the class is  
3 assigned to the initializer field of the class to indicate that the class is the  
4 bootstrap class whose initialization is not triggered by a class initialization barrier.

1 24. The computer-readable storage medium of claim 18, the method  
2 further comprising:  
3 if the class does not have an initialization sequence,  
4 setting the class to a fully initialized state upon the class  
5 being loading by the task, and  
6 assigning the initializer field of the class to a pointer to a  
7 runtime data structure representing the shared part of the class.

1 25. The computer-readable storage medium of claim 18, the method  
2 further comprising instructing the dynamic compiler not to generate native code  
3 for the class initialization barrier of the program method being compiled if the  
4 class targeted by the class initialization barrier is equal to the class that defines the  
5 program method being compiled.

1 26. The computer-readable storage medium of claim 25, the method  
2 further comprising instructing the dynamic compiler not to generate native code

3 for the class initialization barrier of the program method being compiled if the  
4 class targeted by the class initialization barrier is a superclass of the class that  
5 defines the program method being compiled.

1 27. The computer-readable storage medium of claim 26, wherein the  
2 value of the holder of the class initialization barrier is a pointer to a runtime data  
3 structure representing the shared part of the class that defines the program method  
4 that holds the class initialization barrier.

1 28. The computer-readable storage medium of claim 27, the method  
2 further comprising instructing the dynamic compiler not to generate native code  
3 for the class initialization barrier of the program method being compiled if the  
4 value of the initializer field of the class targeted by the class initialization barrier  
5 is:

6 different from the value that indicates that the class is not bootstrap class,  
7 and

8 different from the pointer to the runtime data structure representing the  
9 shared part of the class that defines the program method being compiled.

1 29. The computer-readable storage medium of claim 26, wherein the  
2 value of the holder of the class initialization barrier is a pointer to the shared  
3 runtime data structure representing the program method that holds the class  
4 initialization barrier.

1 30. The computer-readable storage medium of claim 29, the method  
2 further comprising instructing the dynamic compiler not to generate native code  
3 for the class initialization barrier of the program method being compiled if the



4 value of the initializer field of the class targeted by the class initialization barrier  
5 is:  
6 different from the value that indicate that the class is not the bootstrap  
7 class, and  
8 different from the pointer to the shared runtime data structure representing  
9 the program method being compiled.

1 31. An apparatus to indicate to a dynamic compiler of a multitasking  
2 virtual machine when the dynamic compiler can skip generating native code for a  
3 class initialization barrier when compiling a program method, wherein a runtime  
4 representation of classes by the multitasking virtual machine includes a shared  
5 runtime data structure that is shared by multiple tasks, and wherein a native code  
6 produced by the dynamic compiler extends the shared runtime data structure  
7 representing classes and can be executed serially or concurrently by multiple tasks  
8 of the multitasking virtual machine, comprising:  
9 an augmenting mechanism that is configured to augment the shared  
10 runtime data structure representing a shared part of a class with an initializer field;  
11 and  
12 a determining mechanism that is configured to use the initializer field of  
13 the class to determine whether a platform-independent instruction of the program  
14 method may trigger an initialization of the class.

1 32. The apparatus of claim 31, further comprising an initializing  
2 mechanism that is configured to initialize a bootstrap class, wherein the bootstrap  
3 class is initialized during startup of a task of the multitasking virtual machine, and  
4 before any concurrency, due to creation of multiple threads of control within the  
5 task, takes place.

1           33.     The apparatus of claim 32, further comprising an assigning  
2     mechanism that is configured to assign a value of an initializer of the class when  
3     the class is fully initialized, wherein the value includes an indication that one of:  
4           the class is not the bootstrap class,  
5           the class is the bootstrap class and the value of the initializer identifies a  
6     holder of the class initialization barrier that triggered the initialization of the class,  
7     and  
8           the class is the bootstrap class and the value of the initializer further  
9     indicates that the class initialization was not triggered by a class initialization  
10    barrier.

1           34.     The apparatus of claim 33, further comprising:  
2           a setting mechanism that is configured to set a binary variable to zero upon  
3     starting the multitasking virtual machine;  
4           wherein the setting mechanism is further configured to set the binary  
5     variable to one when all bootstrap classes have been initialized by a first task  
6     executed by the multitasking virtual machine;  
7           whereby the binary variable indicates to the multitasking virtual machine  
8     whether all bootstrap classes have been initialized.

1           35.     The apparatus of claim 34, further comprising:  
2           an examining mechanism that is configured to note the holder of the class  
3     initialization barrier; and  
4           wherein the assigning mechanism is further configured to assign the  
5     holder to the initializer field only if the binary variable is zero.

1           36.     The apparatus of claim 33, wherein the assigning mechanism is  
2 further configured to assign the initializer field of the class to a constant value,  
3 wherein the constant value is distinguishable from all other possible values for the  
4 initializer field.

1           37.     The apparatus of claim 36, wherein the constant value is a NULL  
2 pointer.

1           38.     The apparatus of claim 33, wherein a pointer to a runtime data  
2 structure representing the shared part of the class is assigned to the initializer field  
3 of the class to indicate that the class is the bootstrap class whose initialization is  
4 not triggered by a class initialization barrier.

1           39.     The apparatus of claim 33, further comprising a setting mechanism  
2 that is configured to set the class to a fully initialized state upon the class being  
3 loading by the task; and  
4           wherein the assigning mechanism is further configured to assign the  
5 initializer field of the class to a pointer to a runtime data structure representing the  
6 shared part of the class.

1           40.     The apparatus of claim 33, further comprising an instructing  
2 mechanism that is configured to instruct the dynamic compiler not to generate  
3 native code for the class initialization barrier of the program method being  
4 compiled if the class targeted by the class initialization barrier is equal to the class  
5 that defines the program method being compiled.

1           41.     The apparatus of claim 40, wherein the instructing mechanism is  
2 further configured to instruct the dynamic compiler not to generate native code for  
3 the class initialization barrier of the program method being compiled if the class  
4 targeted by the class initialization barrier is a superclass of the class that defines  
5 the program method being compiled.

1           42.     The apparatus of claim 41, wherein the value of the holder of the  
2 class initialization barrier is a pointer to a runtime data structure representing the  
3 shared part of the class that defines the program method that holds the class  
4 initialization barrier.

1           43.     The apparatus of claim 42, wherein the instructing mechanism is  
2 further configured to instruct the dynamic compiler not to generate native code for  
3 the class initialization barrier of the program method being compiled if the value  
4 of the initializer field of the class targeted by the class initialization barrier is:  
5           different from the value that indicates that the class is not bootstrap class,  
6 and  
7           different from the pointer to the runtime data structure representing the  
8 shared part of the class that defines the program method being compiled.

1           44.     The apparatus of claim 41, wherein the value of the holder of the  
2 class initialization barrier is a pointer to the shared runtime data structure  
3 representing the program method that holds the class initialization barrier.

1           45.     The apparatus of claim 44, wherein the instructing mechanism is  
2 further configured to instruct the dynamic compiler not to generate native code for

3 the class initialization barrier of the program method being compiled if the value  
4 of the initializer field of the class targeted by the class initialization barrier is:  
5 different from the value that indicate that the class is not the bootstrap  
6 class, and  
7 different from the pointer to the shared runtime data structure representing  
8 the program method being compiled.